Road Network Detection using Probabilistic and Graph Theoretical Methods

Cem Ünsalan, Beril Sirmacek Member, IEEE,

Abstract

Road network detection from very high resolution satellite and aerial images has diverse and important usage areas such as map generation and updating. Although an expert can label road pixels in a given image, this operation is prone to errors and quite time consuming. Therefore, an automated system is needed to detect the road network in a given satellite or aerial image in a robust manner. In this study, we propose such a novel system. Our system has three main modules as: probabilistic road center detection, road shape extraction, and graph theory based road network formation. These modules may be used sequentially or interchangeably depending on the application at hand. To show the strengths and weaknesses of our system, we tested it on several very high resolution satellite (Geoeye, Ikonos, Quickbird) and aerial image sets. We compared our system with the ones existing in the literature. We also tested the sensitivity of our system to different parameter values. Obtained results indicate that our system can be used in detecting the road network on such images in a reliable and fast manner.

Index Terms

Aerial images, satellite images, edge detection, kernel based density estimation, binary balloon algorithm, graph representation, road network detection.

I. INTRODUCTION

Road network detection from a satellite or an aerial image is an important and challenging remote sensing problem. Its solution may lead to automated map generation and updating which may affect various applications in a positive manner. Also, the solution may help automated path planning for unmanned aerial vehicles. The resolution of recent satellite (like Geoeye, Ikonos, and Quickbird) and aerial images allows road network detection. However, well-known computer vision techniques are not suitable alone for automatically detecting the road network from a given image due to several constraints. Road segments in the image may have different
intensity values. Their widths may change. Moreover, junctions of unknown number of roads and roundabouts may increase the difficulty of the problem. Roads can be occluded by other nearby objects like buildings and trees. Since the resolution of the recent satellite images is fairly high, even vehicles may partially occlude the roads. Therefore, advanced methods are needed to extract the road network from very high resolution satellite or aerial images.

In the literature, there are several methods to detect the road network from a given satellite or aerial image. Three excellent papers by different authors survey and classify road detection methods till 2003 [6], [23], [38]. One set of recent studies focus on straight line based methods for road detection. Katartzis et al. [18] in their work first applied local analysis using morphological filters to detect straight lines. They also used line tracking methods for this purpose. Using global analysis and Markov Random Fields, they combined road segments. Shi and Zhu [29] focused on detecting urban road segments. They obtained a binary image using thresholding to detect line segments. Their method depends on line segment matching and morphological operations. Stoica et al. [37] used a Gibbs point process based method to detect road segments. They modeled the road network as a connection of line segments. Gamba et al. [14] extracted linear features from optical and SAR images using multiple directional filters. They used perceptual grouping to detect road segments with these features. Yang and Wang [42] developed a method to detect main roads from satellite images. First, they detected road primitives such as straight lines and homogenous regions for this purpose. Then, they linked these detected primitives. In the present study, we neither depend on linear features nor straight line segments.

Another set of studies focus on graph representation in road network extraction. Steger et al. [36] worked on complete road network extraction from aerial images. They emphasize that the low level road extraction methods will be fragmented and contain false alarms. Therefore, they benefit from graph theory to model the road network. To handle uncertainties, they used the fuzzy set theory. They used Steger’s method [35] to extract the road segments in their experiments. In a following study, Wiedemann and Hinz [41], proposed to use multispectral images by extracting lines (again using Steger’s method) from all available channels. They use the union operation to have a unique representation. In modeling the road network, they benefit from local, regional, and global information. Wiedemann and Ebner [40] proposed a method to complete incompletely extracted road networks. To do so, they introduced a link hypothesis based on the network characteristics.
Hinz and Wiedemann [15] proposed a general idea on the internal evaluation of object extraction systems. They picked two road extraction methods as examples. They showed that by introducing internal evaluation methodology, the performance of these systems increase. Mayer et al. [21] compared six road extraction methods on six test images (three aerial and three multispectral Ikonos). They aimed to show the potential of these methods on road center extraction. In Section V-E, we compare our method with these methods using the same Ikonos images. All these methods depend on the previously extracted road center pixels. Besides they do not extract the road shape. In this study, we propose a novel road center extraction method. We also propose a novel road shape extraction method. Besides, these two methods interact with each other. Therefore, although our method uses graph formalism in modeling the road network, it is different from the previous work in the literature.

Another set of road detection studies benefit from multispectral information. Ünsalan and Boyer [38] proposed a method to detect the street network and buildings in residential regions using multispectral information in a novel way. Jin and Davis [17] used two different methods to detect roads in urban and rural regions. Their first method depends on segmentation and grouping. Their second method is based on multiscale curvilinear structure extractors. They also benefit from directional morphological filters and multispectral information. Wang et al. [39] detected road segments in panchromatic and multispectral Quickbird images. Mena and Malpica [24] used color information in terms of RGB and HSI color spaces. First, they smoothed the image by median filtering. Then, they applied texture analysis to segment the image. They finally simplified the result by morphological filtering and thinning. Christophe and Inglada [11] combined the multispectral information by spectral angle in their method. They obtained straight line segments by using the gradient direction. However, they need a seed pixel labeled by the user. Zhang and Couloigner [43] benefit from the multispectral information to detect road segments and parks. In this study, we do not benefit from color and multispectral information.

Also there are several studies attacking the road detection problem from different perspectives. Pandit et al. [26] benefit from multitemporal images for road detection. Different from previous studies, they first detect vehicles on the road. Then, they take these as seed points and detect the road network. Unfortunately, their method depends on multitemporal information. Hu et al. [16] defined the pixel footprint by homogeneous polygonal areas around each pixel. Using
Fourier shape descriptors, they classified the road area. In the present study, we do not have a classifier which needs to be trained. Peng et al. [27] worked on updating outdated road maps. They used a multiscale statistical method. They included the GIS information as prior information to their system. In our system we do not need such a prior information. Peteri and Ranchin [28] proposed a modular road shape extraction method. Their first module uses graph representation. The graph can come from a road database or be extracted automatically. Their second module is on road shape extraction by active contours. Being modular, this work resembles our system. However, we propose novel road center extraction and shape extraction methods in this study. Movaghati et al. [25] used particle and Kalman filters interchangeably for road tracking. Unfortunately, their method needs a starting point. In a previous study, we proposed a method to detect the road center pixels which may be taken as the starting point of the present study [32].

In this study, we propose a novel modular system to detect the road network in very high resolution panchromatic satellite and aerial images. In our system, we do not benefit from prior information such as GIS data. Our focus here is extracting road networks with minimal prior information. This is a perfect example for road network extraction of another country, where the prior information is either limited or it does not exist. Our system is composed on three main modules. In the first module, we extract edge pixels as primitives. Then, we use these to detect road centers by a novel probabilistic method. In the second module, we extract the road shape by a binary balloon algorithm. In the third and the final module, we benefit from graph theory to represent the extracted road segments in a graph formation. This allows us to refine and improve the detection results. We applied our road network detection method on several satellite (Geoeye, Ikonos, and Quickbird) and aerial images. These represent different road formations and region properties. We provide the obtained results in the experiments section. In the same section, we compare our method with the ones in the literature from different perspectives. We also test the sensitivity of our system to the used parameters in the experiments section. Next, we explain our system in detail.

II. Probabilistic Road Center Detection

The first module in our system is road center detection. In this module, we pick edge pixels as observations of an unknown probability density function (pdf). Inspired from our previous study,
we benefit from a modified form of variable kernel based density estimation for this purpose [33]. Different from our previous work, we embed the shape constraints of road segments in estimating the unknown pdf. We then apply novel methods to locate the road center pixels from the estimated pdf. In the following sections, we explain each step in detail.

A. Finding Road Primitives

Our road network detection system depends on the edge information in the image. Unfortunately, very high resolution satellite and aerial images may contain undesired details besides road pixels. Hence, standard edge detection methods may not work properly. To overcome this problem, we first apply nonlinear median filtering to smooth the given image. In our previous studies, we also benefit from such nonlinear smoothing operations [31], [33]. After smoothing, we ask the user to indicate whether to find ‘bright’ or ‘dark’ road segments. Unfortunately, edge detection does not work properly for dark road segments. Therefore, this is the only step we ask the guidance of the user. If the road to be detected is dark, then we obtain the negated image and process it.

After median filtering and image negation (if necessary), we apply the Canny edge detector to extract edge pixels as primitives [8]. We picked this edge detector due to its scalability and robustness to noise. We adjusted its parameters for an average test image and fixed them through our experiments. In Section V, we discuss the effect of these parameters on the final road detection results. We also tested several other edge detectors in Section V to show that our system does not depend on a specific edge detector.

We then group the detected edge pixels by their connectedness using connected components analysis [34]. Based on this operation, we first eliminate groups having closed or curved shapes (since they have low probability to represent a road segment). In doing this, we benefit from the shape information as the ratio of the area (sum of pixels for an open curve, sum of all pixels within a closed curve) and the bounding box (minimum rectangle covering the curve) of the group. If this ratio is greater than 0.1 for a group, we discard it. In Section V, we justify this constant threshold value experimentally. After this operation, we have refined edge pixels as \((x_i, y_i)\) for \(i = 1, ..., I\), \(I\) being the total number of refined pixels. We also calculate the gradient direction, \(\theta_i\), for each refined edge pixel.

In Fig. 1, we provide the third Ikonos test image, having a size of 693 \times 1051 pixels. We
will use this image throughout this study to visualize the computation steps. In this figure, we provide the Canny edge detection results as well as the refined edge pixels based on the shape criteria. As can be seen, by using the shape criteria most of the edges belonging to non-road regions are eliminated.

![The third Ikonos test image](image1)

![Canny edges](image2)

![Refined edges](image3)

Fig. 1. The third Ikonos test image, its Canny edges, and refined edges by the shape criteria.

After refining edge pixels based on the shape criteria, we obtain the approximate road width for each edge pixel at \((x_i, y_i)\). To do so, we obtain the nearest road pixel from other groups in terms of the Euclidean distance. We assume this distance as the approximate road width \((w_i)\) for the edge pixel \((x_i, y_i)\). To prevent wrong approximations, if \(w_i\) is higher than 20 pixels, we assume that no nearest edge group is detected. In this case, we set \(w_i\) value as the mean of the previously detected approximate road widths. This way, we can also calculate an approximate road width for road segments represented by one side of their borders. Therefore, missing edges
of a road segment are also taken into account by this method. At the end of the primitive extraction step, we have three sets of values as: road edge pixel locations, \((x_i, y_i)\), the gradient direction, \(\theta_i\), and the approximate road width, \(w_i\), associated with them.

B. Kernel based Density Estimation

Edge pixels extracted in the previous section lead to road center pixel detection. We formulate this step with a probabilistic framework. To do so, we represent possible road center locations as discrete joint random variables. We then estimate their pdf by taking the extracted edge pixels as observations. Each observation may represent a nearby road center based on the road width at that location. This method is an extension of our previous study on building detection [33]. Here, we expand it by embedding the generic road segment shape in estimating the pdf.

For completeness, we start explaining the variable kernel based density estimation as explained in [30]. To represent road segments, we define a fixed kernel which is an elongated Gaussian function as

\[
N(x, y) = \frac{1}{\kappa} \exp \left( -\left( x^2 + \left( \frac{y}{5} \right)^2 \right) \right)
\]

where \(\kappa\) is the normalizing constant. Here, we obtain the elongated Gaussian by scaling the \(y\) coordinate by five. This kernel function satisfies the two constraints required by kernel based density estimation as, \(\sum_x \sum_y N(x, y) = 1\) and \(N(x, y) \geq 0 \ \forall (x, y)\). We specifically picked such a kernel function to model the road characteristics as an ensemble of elongated shapes. Therefore, we embed the generic road shape in density estimation.

In the next phase, we estimate the location of road center pixels based on our previous observations: road edge pixels \((x_i, y_i)\), the road width \(w_i\) (half of this value is used in estimating the road center), and the gradient direction \(\theta_i\) (the angle between the major axis of the elliptical Gaussian with the horizontal axis) associated with each edge pixel. Based on these, the kernel for the \(i\)th observation becomes

\[
N_i(x, y) = \frac{1}{\kappa} \exp \left( A^T A \right)
\]

where
\[
A = \begin{bmatrix}
w_i/2 & 0 \\
0 & w_i/10
\end{bmatrix}
\begin{bmatrix}
\cos(\theta_i) & -\sin(\theta_i) \\
\sin(\theta_i) & \cos(\theta_i)
\end{bmatrix}
\begin{bmatrix}
x - x_i \\
y - y_i
\end{bmatrix}
\] (3)

In Eqn. 3, the matrix \( A \) is composed of the multiplication of scaling, rotation, and shifting parts. Using the kernel given in Eqn. 2 for \( i = 1, ..., I; \) (\( I \) being the total number of refined edge pixels, mentioned in Section II-A) we can estimate the bivariate pdf representing possible road centers as

\[
p(x, y) = \frac{1}{\kappa} \sum_{i=1}^{I} N_i(x, y)
\] (4)

In Fig. 2, we provide the estimated pdf for the third Ikonos test image. This image is color coded (red represent very high values, blue represent low values). As can be seen, even this result is very informative for a human operator since it highlights road segments fairly well.

Fig. 2. Estimated pdf for the third Ikonos test image.

C. Road Center Detection

Based on the estimated pdf in Eqn. 4, we can detect road center pixels. Therefore, we use a two level (high, low) hysteresis thresholding inspired by Canny’s method [8]. To calculate the high threshold value in an adaptive manner, we first obtain the maximum value of the pdf as \( p_m = \max_{(x,y)} \{ p(x,y) \} \). Then, high threshold will be adjusted as \( 0.3 \times p_m \). We take pixel locations higher than this value as possible road centers. In Section V, we justify this multiplier in high threshold calculation. We then embed the road segment characteristics into the detection
process using the following observation. Road segments are continuous and they do not end suddenly. Therefore, some locations close to the detected road center pixels may also be road centers. To handle these, we set the low threshold value in an adaptive manner as $0.01 \times p_m$. We label peak locations higher than this low threshold as possible road centers if their neighbor is also a road center. We provide the detected road center pixels for the third Ikonos test image in Fig. II-D. As can be seen, most road center pixels are successfully detected by this operation.

D. Initial Tracking

To obtain the maximum information from the estimated pdf, we finally apply a basic tracking operation on it. Therefore, we first obtain the endpoints (having just one neighbor) of the detected road center pixels. From each endpoint, we apply a basic tracking as follows. In each iteration, the next possible road pixel is selected from the next three neighbors (lying perpendicular to the endpoint and having the highest pdf value). We iterate until the next candidate pixel falls below a certain threshold. Although this is a very primitive method for tracking, it serves our purpose. We provide the detected road center pixels for the third Ikonos test image after initial tracking in Fig. II-D. As can be seen, most road center pixels are successfully detected. However, the road centers in the upper right portion of the image could not be detected since the road network there is not visible enough. In the next section, we will apply a more advanced tracking method based on the parametric representation of road center pixels.

![Detected road center pixels for the third Ikonos test image before and after initial tracking.](image)

---

October 20, 2011 DRAFT
III. ROAD SHAPE EXTRACTION

The detected road center pixels may not be sufficient alone to the user. The next level of information to be added is the road shape. Therefore, in this section we improve and generalize our previous binary balloon algorithm to extract the shape of road segments [38]. To note here, the term ‘binary balloon’ is used since the method is based on the edge information and is different from the method presented in [12]. This shape information also helps us to refine road detection results. We explain this method next.

A. Binary Balloons

In a previous study, we introduced a binary balloon algorithm to extract road segments and separate houses from binary images (obtained from thresholding multispectral image representations) [38]. Our algorithm was based on initially detected straight line segments (as initial balloons) from binary images. Since we were dealing with a thresholded image, we extracted four set of balloons to represent it. Then, we refined these by a voting scheme. In this study, we generalize and improve this approach in two ways. First, our new binary balloon algorithm only depends on the edge information. Hence, we do not need a thresholded image (based on multispectral information). Second, we do not assume any straight line segments as initial balloons. This leads to more flexible balloon shape extraction without using four set of balloons and the voting scheme. Therefore, this new method is more general and fast compared to the previous one.

To extract the road shape, we pick the road center pixels detected in the previous section as initial balloons. Therefore, we first represent them in a binary matrix form $R$ such that the pixel value is one if there is a road center there. Otherwise, the entries will have value of zero for this matrix. We discard the junction pixels (having more than two neighbors) in this matrix. We apply connected components analysis and group these road segments as $R_k$ for $k = 1, ..., K$. These will serve as initial balloons. For each road segment (initial balloon), we also extract the endpoints as $d^1_k$ and $d^2_k$. Then, we iteratively apply dilation and intersection operations to each road segment separately to obtain the corresponding road shape.

To explain this method in detail, let’s start with the definition of the translation and dilation operations. The translation of a set $R_k$ by $p$ is
\[(R_k)_p = \{c | c = a + p, a \in R_k \}\]  

(5)

The dilation of set \(R_k\) by set \(M\) (the structuring element) is

\[R_k \oplus M = \bigcup_{b \in M} (R_k)_b\]  

(6)

To extract the road shape, we assign a 3 \(\times\) 3 mask composed of all ones to the set \(M\).

For each road segment, \(R_k\), we apply the following iterative operation to extract its shape. We first dilate the corresponding endpoints as \(d^1_k = k^1_k \oplus M\) and \(d^2_k = d^2_k \oplus M\). Then, we dilate the initial road segment and exclude the endpoints as

\[S_k = (R_k \oplus M) \cap (d^1_k \cup d^2_k \cup R_k)'\]  

(7)

Therefore, in the first iteration, we obtain the boundary pixels. For the remaining iterations, we have

\[S_k = (S_k \oplus M) \cap (d^1_k \cup d^2_k \cup R_k)'\]  

(8)

At the end of each iteration, we obtain two separate sets representing the two sides of the road segment.

We apply this operation till we satisfy the stopping criterion as follows. We apply the intersection operation to check whether we have reached the boundary. The boundary is composed of the edge pixels at the beginning of the operation as \(D\). Therefore, for each road side \((S^1_k\) and \(S^2_k\)) we check if the number of pixels \(D \cap S^1_k\) and \(D \cap S^2_k\) are above the 2\% of the total number of pixels in \(R_k\). In Section V, we justify this constant threshold value experimentally.

We continue the iteration until we satisfy this constraint. At the end of the iterations, we obtain the road shape, \(S_k\), for the road segment at hand. We apply this iterative operation for each road segment separately.

B. Shape based Refinement

As we apply the shape extraction operation, some balloons may expand above the limit. This means that, they do not represent a reliable road shape. Or the initial road segment extracted is
not reliable. Therefore, we discard that segment and the balloon extracted from it. We perform this operation by checking the iteration number.

**C. Intensity based Refinement**

The extracted shape also allows us to check whether the road segment belongs to the road network or not. We perform this by the help of the intensity information. If the intensity of the road segment (balloon) is not consistent with the general intensity value of the extracted road network, we discard it. For this purpose, we calculate the median intensity value on all extracted road segments. If the road segment at hand has the median intensity below 80% of this value, we assume it to be an outlier and discard it.

After eliminating the road segments based on the above two criteria, we combine the remaining segments. Finding their boundary, we form the road shape. For future operations, we also keep the road center pixels which have passed the shape and intensity based refinement tests. We provide the extracted road network shape before and after initial tracking in Fig. 4. As can be seen, the extracted road shape for the road network in the third Ikonos test image is fairly good.

![Fig. 4](image)

**Fig. 4.** The third Ikonos test image, road network shape extraction results before and after initial tracking.

**IV. GRAPH THEORY BASED ROAD NETWORK FORMATION**

In previous sections, we detected possible road center pixels. Then, we extracted the corresponding road shape. Besides these, the road network has certain structural (neighborhood and shape) properties. To benefit from this information, we benefit from graph formalism in this
section. Therefore, we first represent the detected road center pixels in a parametric curve form. This allows us to use a parametric tracking method. Then, we form a graph to represent the road network. This leads to refining some road segments due to their shape and neighborhood conditions. Therefore, we can obtain a more robust and correct road network representation. In forming the graph, we benefit from tools in our previous study [38]. However, due to the character of the problem at hand, we extended and improved them substantially.

A. Forming the Graph

In forming the graph, we start with labeling the junctions in road center pixels. We group the remaining pixels by their connectedness. Assume that we obtain $C$ of such pixel groups. We fit a parametric curve for each group as

$$ (x_c(t), y_c(t)) = \left( \sum_{l=0}^{L-1} a_l t^l, \sum_{l=0}^{L-1} b_l t^l \right) $$

for $c = 1, \ldots, C$ curves. $t = [1, \ldots, n_c]$ stands for the parameter value. Polynomial coefficients $a_l$ and $b_l$ are obtained from least square fitting. In Eqn. 9, $L$ is the fit order of the parametric curve picked as $L = 0.1 \times n_c$ to have an adaptive fit order.

We represent the parametric curves and junctions in a graph form. A simple graph $G$ consists of a finite non-empty set $V$ and a set $E$ of two-element subsets of $V$. The set $V$ is called the vertex set of $G$; the set $E$ is called the edge set of $G$. We write $G = (V, E)$, to denote the graph $G$ with vertex set $V$ and edge set $E$. In forming the road network, we assign each junction (extracted in the previous step) as the vertex of the graph as $v_j$. We assign the parametric curves, $(x_c(t), y_c(t))$ for $c = 1, \ldots, C$ as the edges of the graph. We provide the graph formed from the third Ikonos test image in Fig. IV-C2.

B. Parametric Tracking

In this tracking phase, we benefit from the parametric curves (edges of the graph) extracted in the previous section. Since we have the parametric representation of a curve, we can estimate the next point on its free end (the junction of the edge which has no other edges connected to it). To do so, we first increase the parameter value of the curve above its limits. Then, by calculating the polynomial values there, we estimate the next coordinate of the curve. We also
pick the two neighbors of this point lying in the orthogonal direction to the parametric curve. We calculate the pdf values of these three points from Eqn. 4 and pick the one having the maximum value. If this value is higher than \(0.05 \times p_m\), we update the parametric curve representation to include this new entry. In Section V, we justify this constant in thresholding experimentally. We iterate this process until there is no new point to be added. We apply this method to all vertices having free endpoints. At the end of this operation, we reform the graph since some edges and vertices may be changed. We provide the reformed graph for the third Ikonos test image after the parametric tracking operation in Fig. IV-C2. We also apply the shape extraction module to the results obtained by the graph formalism. We provide the road detection results in Fig. IV-C2. As can be seen, some missing road segments in the initial graph have been extracted here.

C. Refining the Graph

Finally, we have the full road network at hand. However, some edges may not reliably represent a road segment. We eliminate these by embedding the structure information on the graph by unary and binary constraints. Unary constraints are based on the properties of each graph edge alone. Binary constraints take the neighborhood information in the graph into account.

To define unary and binary constraints, for each edge, we define three measures. The first is arclength, defined as

\[
\gamma_c = \sum_{t=1}^{n_c-1} \sqrt{(x_c(t+1) - x_c(t))^2 + (y_c(t+1) - y_c(t))^2}
\]  

The second measure is straightness defined as

\[
\rho_c = \frac{\sqrt{(x_c(n_c) - x_c(1))^2 + (y_c(n_c) - y_c(1))^2}}{\gamma_c}
\]  

This measure has the value one for straight lines. It has the value zero for closed shapes since the starting and the end point is the same. Finally, the third measure is the angle values (for the two endpoints), \(\alpha_c^1\), \(\alpha_c^2\) as

\[
\alpha_c^1 = \arctan \left(\frac{y_c(1) - y_c(2)}{x_c(1) - x_c(2)}\right)
\]  

\[
\alpha_c^2 = \arctan \left(\frac{y_c(n_c) - y_c(n_c-1)}{x_c(n_c) - x_c(n_c-1)}\right)
\]
1) **Unary Constraints**: We have four unary constraints to refine the graph. **If an edge is long and straight, we should keep it in the graph since it resembles a road segment. If the line is not straight, unary constraints will not work.** Therefore, in implementation, we added tolerances. In other saying, if $\gamma_c > 20$ and $\rho_c > 0.3$ for the edge, we keep it. If an edge is short and curved, it may not represent a road segment reliably. Therefore, we should discard it. Based on this constraint, we discard edges with $\gamma_c < 15$ and $\rho_c < 0.2$. Moreover, since a road segment can not be circular under normal conditions, we discard edges with $\rho_c < 0.1$ independent of their arclength. Finally, if an edge has only one neighbor and is short ($\gamma_c < 15$), probably it belongs to a house nearby. Therefore, we also discard it.

2) **Binary Constraints**: To refine the graph further, we apply two more binary constraints. Here, we apply these on edges not discarded by unary constraints. Assume that we have two edges $e_i$ and $e_j$. If $\gamma_i + \gamma_j > 15$, $|\rho_i - \rho_j| < 0.3$ and $|\alpha_1^i - \alpha_2^j| < \pi/3$ these two edges may represent a longer road segment. Therefore, we keep them in the graph. We apply this test to three edge pairs also. Finally, if an edge lies between two long edges, we keep it since it connects these long edges.

Both unary and binary constraints refine the graph such that, strong edges are kept in the graph and the remaining problematic edges are discarded. We provide the final graph after applying unary and binary constraints in Fig IV-C2. As can be seen, some problematic edges are eliminated.

We finally apply the shape extraction module after refining the graph using unary and binary constraints. We provide the road detection results in Fig. IV-C2. As can be seen, the road network extraction in this step is far better than previous steps.

V. **Experiments**

To show the strengths and weaknesses of our road network detection system, we perform several tests in this section. Our test bed is composed of large and representative images from four different sensors as Geoeye, Ikonos, Quickbird satellite and aerial. These test images are obtained from different sources. Unfortunately, our Ikonos and aerial test images are licensed and can not be shared. However, the Geoeye and Quickbird test images are publicly available and can be obtained from [2], [3], [5]. As for the quality, Ikonos and Geoeye image sets are
Fig. 5. The graph extraction steps for the third Ikonos test image.

(a) Initial graph
(b) After parametric tracking
(c) After unary and binary constraints

Fig. 6. The third Ikonos test image, road network shape extraction results.

(a) After parametric tracking
(b) After unary and binary constraints
original. On the other hand, Quickbird and aerial image sets are jpeg compressed. Therefore, their quality is not as good as the original versions.

The resolution of our Geoeye, Ikonos, Quickbird, and aerial images are 0.4 m, 1 m, 0.6 m, and 0.6 m respectively. To note here, due to the viewing angle of the satellite, while capturing the image, these values may fluctuate slightly. Since our test images have different resolutions, we downsampled Geoeye, Quickbird, and aerial images by 50% to have one set of parameters through our system. Decreasing the resolution helped us in two ways. First, the computation time of the algorithm is proportional to the size of the test image. Therefore, decreasing the image size also decreased the computation time of our algorithm. Second, although the details present in high resolution images may be useful for object detection, they may act as noise for road extraction. Therefore, it is best to find an optimal image resolution for road extraction.

Our test images contain diverse road characteristics both in grayscale and shape. Some of our images represent a very complex road network close to city centers. Some other test images represent the road network in rural regions, where the land formation may be misleading. As a remainder, road characteristics do not change frequently in city centers. On the other hand, around rural regions and city borders, road networks expand and change most. Therefore, detecting the road network in these regions is more important.

In quantifying the performance values, we follow the criteria used by Peng et al. [27] as

\[
\text{Completeness} = \frac{TP}{TP + FN} \tag{14}
\]

\[
\text{Correctness} = \frac{TP}{TP + FP} \tag{15}
\]

\[
\text{Quality} = \frac{TP}{TP + FP + FN} \tag{16}
\]

where \(TP\), \(FN\), and \(FP\) stand for true positive, false negative, and false positive respectively. The three criteria given in Eqn. 14 has the following explanations. The \textit{completeness} value indicates the percentage of the ground truth road pixels detected. The \textit{correctness} value indicates the percentage of the correct road pixels. Finally, the \textit{quality} value indicates the goodness of the result. In the following sections, we will call these as \textit{Comp}, \textit{Corr}, and \textit{Qual} respectively. In calculating these values, we used manually formed ground truth data (since any previously
formed ground truth data is unavailable). To avoid any bias, two authors and a PhD student from the German Aerospace Center prepared this ground truth data.

In the following sections, we provide our experimental results on the sensitivity of our system to the edge detector type, robustness to parameter variations, performance, and the computation time of our system modules separately. **We compare our road center detection and road shape extraction modules with the state of the art methods in the literature. We also provide the comparison results of our system with a commercial one in this section.**

A. Tests on Different Edge Detectors

Our road network detection system depends on the edge information. In tests, we picked the Canny edge detector since it has a better response. However, our system also works on different edge detectors such as Sobel, Prewitt, Roberts, Log, and Edison [34], [22]. We pick the third Ikonos test image and provide the performance of our system with different edge detectors in Table I.

**TABLE I**

<table>
<thead>
<tr>
<th>Edge detector</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canny</td>
<td>0.8368</td>
<td>0.8145</td>
<td>0.7028</td>
</tr>
<tr>
<td>Sobel</td>
<td>0.9285</td>
<td>0.6436</td>
<td>0.6132</td>
</tr>
<tr>
<td>Prewitt</td>
<td>0.9319</td>
<td>0.6393</td>
<td>0.6108</td>
</tr>
<tr>
<td>Roberts</td>
<td>0.9696</td>
<td>0.4782</td>
<td>0.4711</td>
</tr>
<tr>
<td>LoG</td>
<td>0.9427</td>
<td>0.6735</td>
<td>0.6470</td>
</tr>
<tr>
<td>Edison</td>
<td>0.9351</td>
<td>0.6777</td>
<td>0.6472</td>
</tr>
</tbody>
</table>

As can be seen, the detection results using different edge detectors (besides Canny) have similar completeness and correctness values. The Canny edge detector differs from the rest in terms of a balanced completeness and correctness value. Based on these results, we can first claim that our system also works on different edge detectors. Second, the Canny edge detector has a better performance in terms of joint completeness and correctness values. Therefore, we picked it in our system.
B. Tests on System Parameters

In our road network detection system, most parameters are adjusted automatically. However, there are also some manually adjusted parameters in the system. In this section, we justify these selections experimentally. We also test the sensitivity of our road network detection system to variations of these parameters. We pick the third Ikonos test image in the following tests.

1) Canny Edge Detector: The first parameter test is on the Canny edge detector. We adjusted its variance parameter automatically. To detect weak edges, we also fixed the low threshold level (in hysteresis thresholding) to 0.01 (for normalized values between zero and one). In Fig. V-B5, we provide the completeness (marked by solid squares), compactness (marked by solid circles), and quality (marked by solid triangles) values for the high threshold (in hysteresis thresholding) levels between 0.15 and 0.35. We follow the same notations in the following parameter tests also. As can be seen, our detection results are similar for this parameter range. Around 0.25, we obtain good results in terms of performance. Therefore, we fixed the high threshold value to 0.25 throughout the tests.

2) Shape based Refinement in Road Primitives: In finding road primitives, we eliminate some edge segments based on their shape values in Section II-A. Here, we change the threshold used in the refinement step (from 0.05 to 0.30) and observe its effect on the final road network detection performance. We provide the results in Fig. V-B5. In this figure, as the threshold value exceeds 0.1, the compactness value stabilizes. Therefore, we fixed this parameter to 0.1 in the experiments.

3) The Constant in Thresholding the pdf in Road Center Detection: The third test is on the constant in thresholding the pdf value discussed in Section II-C. In Fig. V-B5, we provide the final road network detection results as we change this constant between 0.2 and 0.4. As can be seen, the obtained results are fairly stable. However, the completeness and compactness values change slightly as we exceed 0.3. Therefore, we fixed the constant to 0.3 in the experiments.

4) Binary Balloon Expansion Threshold: The fourth test is on the binary balloon expansion threshold as mentioned in Section III-A. We changed this parameter from 0.01 to 0.05 and provide the final road network detection results in Fig. V-B5. As can be seen, the obtained performance values are almost the same. Therefore, we fixed this parameter to 0.02 throughout the experiments.
5) *Parametric Tracking Threshold*: The final test is on the constant in the adaptive threshold value used in parametric road tracking module of our system explained in Section IV-B. We changed this value from 0.04 to 0.08 and provided the results in Fig. V-B5. As can be seen, the performance values do not change much. Therefore, we fixed it to 0.05 throughout the experiments.

**C. Performance Tests**

In this section, we first tabulate the performance of our system to each image set (obtained from a specific sensor). We also provide the best and worst results for each image set in separate figures. Then, we provide the overall performance value. Since we extract different outputs at different modules of our system, we tabulate their performances separately as: initial detection and tracking, graph representation (including parametric tracking), and graph refinement (using unary and binary constraints).

1) *Results on Geoeye Satellite Images*: We picked 12 Geoeye images having a total size of 9454319 pixels. In these images, the road network occupies 1176799 pixels in total. We provide the performance of our system for each module in Table II. As can be seen, the average performance of our system at each module is fairly good.

<table>
<thead>
<tr>
<th>Module</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial detection</td>
<td>0.7805</td>
<td>0.7363</td>
<td>0.6100</td>
</tr>
<tr>
<td>Initial tracking</td>
<td>0.7834</td>
<td>0.7798</td>
<td>0.6415</td>
</tr>
<tr>
<td>Graph representation</td>
<td>0.7842</td>
<td>0.7929</td>
<td>0.6509</td>
</tr>
<tr>
<td>Graph refinement</td>
<td>0.7960</td>
<td>0.7815</td>
<td>0.6511</td>
</tr>
</tbody>
</table>

We provide the extracted road network on two Geoeye test images such that we have the best and worst detection results in terms of the completeness value (using all modules) in Fig. 8. As can be seen, the best result is obtained in a residential region (the fifth Geoeye image). In this image, although the houses are highly populated, the road network can be clearly seen. On the other hand, the variability of the road width in this image is apparent. Our method handled this variability fairly well. The worst result is obtained on the junction of
Fig. 7. Parameter sensitivity tests. Solid squares-completeness, solid circles-compactness, solid triangles-quality.
a residential and an industrial region (the fourth Geoeye image). In this image, the cars on the road are visible. Although these act as noise, they are handled fairly well by our probabilistic detection method. As in the fifth Geoeye image, the road width varies in this image and our method handled it fairly well. The possible reason for the poor performance on the fourth Geoeye image is as follows. In the industrial region, parts of large buildings are also labeled as road segments. Therefore, the false alarm rate increased. To handle such false alarms, the color information may be used in our system in the future.
2) Results on Ikonos Satellite Images: Next, we pick 11 Ikonos images having a total size of 11681756 pixels of which 707811 belong to the road network. We provide the performance of our system on these images in Table III. Similar to the Geoeye images, we obtain fairly good results on the Ikonos image set.

**TABLE III**

**Road network detection performances on 11 Ikonos satellite test images.**

<table>
<thead>
<tr>
<th>Module</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial detection</td>
<td>0.7723</td>
<td>0.6687</td>
<td>0.5586</td>
</tr>
<tr>
<td>Initial tracking</td>
<td>0.7984</td>
<td>0.6700</td>
<td>0.5730</td>
</tr>
<tr>
<td>Graph representation</td>
<td>0.8088</td>
<td>0.6893</td>
<td>0.5927</td>
</tr>
<tr>
<td>Graph refinement</td>
<td>0.8220</td>
<td>0.6804</td>
<td>0.5930</td>
</tr>
</tbody>
</table>

For the Ikonos images, we obtained best and worst results on a residential and rural image respectively as in Fig. 9. **On the residential region (the eight Ikonos image), since the contrast between the road pixels and the environment is large, the detection results are better than the rest.** Although there are nearby houses in this image, they did not affect the road detection results. For the rural region (the tenth Ikonos image), some road segments can be seen clearly. However, some other road segments have grayscale values close to the background. Although our system was able to detect main road segments, it was unable to detect the low contrast and narrow road segments in this image. Besides, the tree formations and geological shapes in this image resemble road segments. These are also possible reasons for the poor performance in the tenth Ikonos image.

3) Results on Quickbird Satellite Images: We next pick nine Quickbird images having a total size of 4353905 pixels of which 195450 are road network pixels. For each module, the performance of our system on these images is given in Table IV. Unfortunately, the performance of our system on Quickbird images are not as good as the Goeye and Ikonos images. As we mentioned previously, these images were not original (jpeg compressed). This may have caused a decrease in performance.

As in the previous image sets, we provide the best and worst results on the Quickbird image set in Fig. 10. **The best result is obtained on a rural region (the fourth Quickbird image).** In this image, the road characteristics are fairly dominant. Besides, the land formations
Fig. 9. The best and worst road network detection results on two Ikonos test images.
TABLE IV
ROAD NETWORK DETECTION PERFORMANCES ON NINE QUICKBIRD SATELLITE TEST IMAGES.

<table>
<thead>
<tr>
<th>Module</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial detection</td>
<td>0.4657</td>
<td>0.7306</td>
<td>0.3974</td>
</tr>
<tr>
<td>Initial tracking</td>
<td>0.5134</td>
<td>0.7556</td>
<td>0.4403</td>
</tr>
<tr>
<td>Graph representation</td>
<td>0.5408</td>
<td>0.7659</td>
<td>0.4641</td>
</tr>
<tr>
<td>Graph refinement</td>
<td>0.5650</td>
<td>0.7488</td>
<td>0.4750</td>
</tr>
</tbody>
</table>

in this region are also fairly smooth. The worst performance is obtained on a rural region (the fifth Quickbird image). This image has characteristics similar to the tenth Ikonos image. Therefore, the same reasons for the poor performance also apply here.

Fig. 10. The best and worst road network detection results on two Quickbird test images.
4) Results on Aerial Images: We next pick 10 aerial images. The total size of these images is 14938006 pixels, 772774 of them representing the road network. We provide the performance of our system on these test images in Table V. Again, the results on these images are not as good as the Geoeye and Ikonos images. To note here, these images were also jpeg compressed.

<table>
<thead>
<tr>
<th>Module</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial detection</td>
<td>0.6093</td>
<td>0.7091</td>
<td>0.4874</td>
</tr>
<tr>
<td>Initial tracking</td>
<td>0.6564</td>
<td>0.7170</td>
<td>0.5213</td>
</tr>
<tr>
<td>Graph representation</td>
<td>0.6825</td>
<td>0.7531</td>
<td>0.5577</td>
</tr>
<tr>
<td>Graph refinement</td>
<td>0.6992</td>
<td>0.7454</td>
<td>0.5644</td>
</tr>
</tbody>
</table>

In the aerial image set, the best performance is obtained on a residential region (the third aerial image) as given in Fig. 11. In this image, the contrast between the road pixels and the background is fairly high. Therefore, our system detected the road network reliably. Although there are nearby buildings in the region, they did not affect the road detection performance much. The only major missing road segment in this image is in the lower part. It seems the Canny edge detector could not detect the borders of this road segment. By manually adjusting the edge detection parameter, this road segment can also be detected. The worst result is obtained on another residential region (the fourth aerial image) as given in Fig. 11. This image is similar to the third aerial image. Again, one branch of the road network could not be detected in this image due to its contrast with the background. Therefore, the performance decreased substantially on this image.

5) The Overall Performance: Finally, we provide the overall performance of our system on all test images. On gross total, the size of our test images becomes 40427986 pixels. The road network in these images is represented by 2852834 pixels. For each module, the performance of our system is given in Table VI. As can be seen, on such a large and diverse image set, the obtained average performance values are fairly good.
Fig. 11. The best and worst road network detection results on two aerial test images.
TABLE VI
ROAD NETWORK DETECTION PERFORMANCES ON ALL TEST IMAGES.

<table>
<thead>
<tr>
<th>Module</th>
<th>Comp</th>
<th>Corr</th>
<th>Qual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial detection</td>
<td>0.6936</td>
<td>0.7116</td>
<td>0.5414</td>
</tr>
<tr>
<td>Initial tracking</td>
<td>0.7230</td>
<td>0.7336</td>
<td>0.5726</td>
</tr>
<tr>
<td>Graph representation</td>
<td>0.7369</td>
<td>0.7542</td>
<td>0.5942</td>
</tr>
<tr>
<td>Graph refinement</td>
<td>0.7523</td>
<td>0.7441</td>
<td>0.5976</td>
</tr>
</tbody>
</table>

D. The Computation Time

We analyze the computational load (in terms of CPU timings) of our system in this section. Therefore, we pick the third Ikonos test image and obtained the CPU timings of our system with all modules used. We used Matlab as the coding environment on a PC having Intel Core2Quad processor with 3.0 GHz clock speed. Since we have four modules, we provide the CPU time needed for each separately as follows. Initial road detection module (including all preprocessing operations) needs 40.19 sec. The initial tracking operation on this module adds an extra 18.94 sec. The next module is graph representation (including parametric tracking) and it needs 52.51 sec. Finally, the graph refinement module using unary and binary constraints need 13.20 sec. The total time needed for all operations sum up to 124.84 sec. As can be seen, the most time consuming module of our system is graph representation. Based on these timings and the corresponding performances, the user may select which modules to use in his or her applications. To note here, this timing may increase or decrease based on the image complexity at hand.

Although this timing depends on the computer used, we believe that this result is comparable with a human expert’s performance. Since our system provides the road shape information, it may even be faster than an expert. Besides, as the computer technology advances the timing advantage will become clearer. It should also be mentioned that, our system may work 24 hours a day which is not the case for a human expert.

E. Comparison with the Existing Methods

In this section, we compare our road network detection system with the similar methods in the literature. Since most existing methods either focus on road center detection or road shape extraction, we perform the comparison in two steps. First, we consider the
road center detection module of our system. Then, we consider the road shape extraction module. In both comparisons, we quantify the results. We provide all the visual comparison results in the attached supplementary file.

1) Road Center Detection Comparison: In comparing the road detection module with the ones in the literature, we applied two tests. First, we tested our system on three multispectral Ikonos test images (kindly provided by Prof. H. Mayer) used in a previous work [21]. In Mayer’s work, the road center pixel detection results are tabulated using six different methods for the same images. Therefore, we were able to compare our road detection module with these quantitatively.

We obtained the grayscale version of the multispectral images and used our road center detection module on these. Although these images have four meter resolution, we were able to run our method on them. We tabulated our results as well as the ones reported in Mayer’s work (in terms of completeness and correctness) in Table VII. In this table, Img-1, Img-2, Img-3 correspond to images Ikonos1-Sub1, Ikonos3-Sub1, and Ikonos3-Sub2 (as called by Mayer) respectively. In this table, we call other methods in the same name with Mayer.

<table>
<thead>
<tr>
<th>Image / Method</th>
<th>Img-1</th>
<th>Img-2</th>
<th>Img-3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Comp</td>
<td>Corr</td>
<td>Comp</td>
</tr>
<tr>
<td>Bacher</td>
<td>0.34</td>
<td>0.66</td>
<td>0.81</td>
</tr>
<tr>
<td>Beumier</td>
<td>0.48</td>
<td>0.69</td>
<td></td>
</tr>
<tr>
<td>Gerke W</td>
<td>0.27</td>
<td>0.41</td>
<td>0.80</td>
</tr>
<tr>
<td>Gerke WB</td>
<td>0.19</td>
<td>0.49</td>
<td>0.68</td>
</tr>
<tr>
<td>Hedman</td>
<td>0.31</td>
<td>0.51</td>
<td>0.77</td>
</tr>
<tr>
<td>Malpica</td>
<td>0.25</td>
<td>0.74</td>
<td>0.60</td>
</tr>
<tr>
<td>Zhang</td>
<td>0.56</td>
<td>0.41</td>
<td>0.72</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.63</td>
<td>0.40</td>
<td>0.73</td>
</tr>
</tbody>
</table>

In providing our results (in the proposed line) in Table VII, we had the same assumptions as Mayer et al. did. In forming the performance table, they mention that the values for completeness and correctness should be at least 0.6 and 0.75, so that the results become...
practical. In accordance with this constraint, our system passes both thresholds for the Ikonos3-Sub1 and Ikonos3-Sub2 images. This is also the case for most previous studies. Different from previous studies, our method also passes the completeness threshold for the Ikonos1-Sub1 image.

If we look more closely in Table VII, we can summarize the following observations. For the Ikonos1-Sub1 image, our method has the highest completeness value with the least correctness. For the Ikonos3-Sub1 image, our method has the fourth best completeness value with the second best correctness. For the Ikonos3-Sub2 image, our method has the fourth best completeness value with the second best correctness. Therefore, our method is comparable (even better for some cases) with the existing methods. Please also remember that, this comparison is only on road center detection. However, our method also provides road shape information which will be considered in the following section.

As a second road center detection test, we pick the commercial Halcon software to extract curvilinear structures from the third Ikonos test image (the benchmark image of this study) [4]. The Halcon software uses Steger’s method [35] in extracting lines. Based on Mayer’s performance criteria, the completeness obtained from the third Ikonos image using Halcon is 0.5601 with the correctness value being 0.8201. On the same image, our system (only using the road detection module) has the completeness and correctness values as 0.7155 and 0.7712 respectively. Comparing both results, we can claim that our completeness value is fairly higher than Halcon’s result, with comparable correctness values.

2) Road Shape Extraction Comparisons: We next compare the road shape extraction module of our system with six different methods in the literature. Five of these methods are active contour and level set based ones introduced by Caselles et al. [9], Chan and Vese [10], Li et al. [20], Lankton and Tannenbaum [19], and Bernard et al. [7]. We used the publicly available software prepared by Bernard [1] for these methods. We also used the mean shift clustering method introduced by Comaniciu and Meer [13] to label road segments. Using all these methods, we provide the road detection (in terms of shape) results on the third Ikonos test image in Table VIII. To eliminate any bias, we provide both our road center detection result and the one provided by Halcon as initial balloons in this table. We used the initial balloons to refine clusters in the mean shift clustering method.

As can be seen in Table VIII, the initial balloon (either provided by our method or
TABLE VIII
Comparison of road shape extraction methods on the third Ikonos image.

<table>
<thead>
<tr>
<th>Road C. / H</th>
<th>Halcon</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Comp</td>
<td>Corr</td>
</tr>
<tr>
<td>Comaniciu</td>
<td>0.4778</td>
<td>0.5632</td>
</tr>
<tr>
<td>Caselles</td>
<td>0.8501</td>
<td>0.6249</td>
</tr>
<tr>
<td>Chan</td>
<td>0.7904</td>
<td>0.6031</td>
</tr>
<tr>
<td>Li</td>
<td>0.4941</td>
<td>0.9820</td>
</tr>
<tr>
<td>Lankton</td>
<td>0.8755</td>
<td>0.5555</td>
</tr>
<tr>
<td>Bernard</td>
<td>0.4161</td>
<td>0.9859</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.8959</td>
<td>0.6556</td>
</tr>
</tbody>
</table>

Halcon) does not affect the road detection performance much for the six methods. Only our method is affected. This is reasonable since our system is optimized within itself. On the other hand, using another road center method in our system still gives fairly good results. This shows the modularity of our road network detection system. In terms of the completeness value, our system gives the third best result after Caselles and Lankton’s methods. Among these three methods, our method gives the best completeness and quality values. In terms of the correctness value, our system gives the third best result after Bernard and Li’s methods. Among these three methods, our method again gives the best correctness and quality values.

On the third Ikonos image, the computation times for the active contour models are as follows (excluding the initial balloon extraction step): Caselles 64.76 sec., Chan 68.91 sec., Li 515.43 sec., Lankton 297.06 sec., Bernard 478.61 sec. As we mentioned in the previous section, the time needed for our method is 84.65 sec. excluding the initial road detection step (to be fair). Again, our method gives the third best timing after Caselles and Chan’s methods. Among these three, our method gives the second best completeness and the best correctness values.

After these extensive comparisons, we can claim that our proposed method has some clear advantages compared to the existing methods in the literature. It also has some shortcomings, which we believe can be eliminated by using extra information (such as multispectral representations). To note here, some of our test images are open to public.
They can be used as standard benchmark images for road detection studies in future studies.

VI. Final Comments

In this study, we propose a novel modular system to detect the road network from a given panchromatic very high resolution satellite or aerial image. We first detect the road center pixels using a novel probabilistic framework. We benefit from the edge information as road primitives. By a variable kernel based density estimation, we locate the possible road center pixels as the first output of our system. The probabilistic method eliminates the need for parallel edges to represent road segments. Therefore, it overcomes the missing data problem. The probabilistic method does not depend on a specific edge detector as shown in Section V-A. Probabilistic representation also simplifies tracking. Although there are various road center tracking approaches in the literature, they most of the times depend on grayscale or multispectral information. These values may change in the image due to intensity changes. However, the probabilistic representation is robust to intensity changes since it depends on the edge information.

Our method does not only extract road center pixels. It also extracts the road shape and road width information via a novel balloon algorithm. Therefore, we do not use any prior information about the road width. This may give information for road classification purposes (i.e. highways, streets, etc.) in future studies. Our balloon algorithm is local, does fitting for each part separately. Therefore, it is more robust. This comes from our probabilistic representation.

We benefit from graph theory to represent the road network in a parametric and structural form. This helps us to refine and improve our road network detection results further. Unlike previous approaches, we do not use edge based graph representation. Instead, the edge information leads to a probabilistic representation. This leads to a graph based method. To note here, our graphs also model the road segments extracted by our novel balloon algorithm. Applying the road shape extraction module to this result, we obtain the final output of our system.

These novel contributions lead to road network detection using only panchromatic images. Unlike the previous approaches, we do not benefit from color information (since
it may not be available) or multispectral data which leads to easy extraction of man-made structures (however, its availability may also be limited). Besides, we do not benefit from any prior information such as GIS data. However, we can embed this information to our method. Our focus here is extracting the road network with minimal prior information. This is a perfect example for road network extraction of another country, where the prior information is either limited or it does not exist.

We tested our system on a diverse data set obtained from Geoeye, Ikonos, Quickbird satellite and aerial images from different regions with fairly diverse road and surface characteristics. We believe that some of our images represent a fairly complex road network, especially close to city centers. The extraction of the road network from the city center is the candidate for the most difficult scenario. However, most road detection methods do not perform well on city centers. For these locations, extra DSM and DTM data is needed. Besides, road characteristics do not change in city centers frequently. On the other hand, around rural regions and city borders, road networks expand and change most. Therefore, detecting the road network in these regions is more important. On all the test images, we obtained fairly good detection results. Since we have tested our system on such a diverse data set, we can claim that our system does not specifically depend on a certain image type.

There are two main reasons for the errors in our system. First, some nearby buildings are detected as road segments. This error can be corrected if a building detection system is used in parallel. Second, there are some inevitable errors due to the road like patterns in farms and rural regions. Using color information may be of help to overcome this problem. There are also some road segments missed by our system. The main reason for these is the low visibility of the segments. Finally, our system does not discriminate rivers since they resemble road segments. However, they can be discriminated by using color information. Besides these acceptable shortcomings, we obtained very promising results on a large and diverse test set.

Our system has several advantages. It does not depend on the user interaction, such as manually labeling the starting point of the road network. It only needs the road intensity label (‘bright’ or ‘dark’) from the user. Most parameters of our system are adjusted automatically. The road detection performance is also fairly insensitive to changes in the

October 20, 2011 DRAFT
remaining parameters. The road detection performance is also fairly insensitive to the image resolution within its limits. In Section V, we tested our system on 0.8 m (after downsampling) to 4 m test images and the system worked fine. However, if we have a test image with less than five meter resolution, our edge detection module will not work reliably. Hence, all the dependent modules will fail. One solution to this problem is adjusting the Canny edge detection parameter manually there.

We also compared our system with the state of the art road detection methods on the same image set. The results show the usefulness of our system. Therefore, we can suggest that our system to be used in detecting the road network from very high resolution satellite and aerial images in a reliable and fast manner.

ACKNOWLEDGMENT

We would like to thank Prof. Dr. Helmut Mayer from Universität der Bundeswehr München for providing us their test data. We would also like to thank PhD candidate Sebastian Türmer from Technische Universität München for obtaining road detection results on the Halcon software.

REFERENCES


